A Novel Learnable Interpolation Approach for Scale-Arbitrary Image Super-Resolution

Jiahao Chao[1] , **Zhou Zhou**[1] , **Hongfan Gao**[1] , **Jiali Gong**[1] , **Zhenbing Zeng**[2*] and **Zhengfeng Yang**[1*]

[1]Shanghai Key Lab of Trustworthy Computing, East China Normal University, Shanghai, China;
[2]Shanghai University of Finance and Economics Zhejiang College, Jinhua, China
{jhchao502, zhouzhou, hfgao, gongjl}@stu.ecnu.edu.cn, zbzeng@shu.edu.cn, zfyang@sei.ecnu.edu.cn

Abstract

Deep convolutional neural networks (CNNs) have achieved unprecedented success in single image super-resolution over the past few years. Meanwhile, there is an increasing demand for single image super-resolution with arbitrary scale factors in real-world scenarios. Many approaches adopt scale-specific multi-path learning to cope with multi-scale super-resolution with a single network. However, these methods require a large number of parameters. To achieve a better balance between the reconstruction quality and parameter amounts, we propose a learnable interpolation method that leverages the advantages of neural networks and interpolation methods to tackle the scale-arbitrary super-resolution task. The scale factor is treated as a function parameter for generating the kernel weights for the learnable interpolation. We demonstrate that the learnable interpolation builds a bridge between neural networks and traditional interpolation methods. Experiments show that the proposed learnable interpolation requires much fewer parameters and outperforms state-of-the-art super-resolution methods.

1 Introduction

The single image super-resolution (SISR) task aims to recover high-resolution images from degraded low-resolution images. To address the SISR task, there are mainly two categories of SISR methods, namely traditional methods [Aghighi, 2015; Akhtar *et al.*, 2015] and deep learning based methods [Dong *et al.*, 2015; Lim *et al.*, 2017; Zhang *et al.*, 2018b; Zhang *et al.*, 2018a; Liang *et al.*, 2021; Liu *et al.*, 2021]. In the last decades, CNN based methods [Lim *et al.*, 2017; Zhang *et al.*, 2018a; Zhang *et al.*, 2018b] have gradually become the mainstream approach for SISR problems. Nevertheless, most of them are designed as scale-specific networks. For real-world applications, it is not uncommon to zoom the image to a customized scale factor instead of a specific scale factor. Thus, it is vital to design scale-arbitrary networks to enable a wider application.

*Corresponding author.

As we know, traditional image interpolation approaches such as nearest-neighbor interpolation, bilinear interpolation, lanczos interpolation and bicubic interpolation [Lin *et al.*, 2008; De Boor, 1962] are naturally capable of addressing the scale-arbitrary super-resolution problem. They map the neighboring pixels to the corresponding target pixels in HR via the selected kernel functions. However, these kernel functions are fixed, leading to mediocre performance.

Several works attempt to address scale-arbitrary super resolution (SR) using deep learning methods. The multi-path learning strategy is widely employed to address super resolution with different scale factors [Lim *et al.*, 2017; Ahn *et al.*, 2018; Wang *et al.*, 2018b]. To be concrete, the principal components of the model (*i.e.*, the feature extraction components) are shared and then scale-specific pre-processing paths and upsampling paths are attached. Meta-SR [Hu *et al.*, 2019] applies meta-learning to tackle the super-resolution task of different non-integer scale factors in a single network. ArbSR [Wang *et al.*, 2021] shares most of the parameters for different scales and handle the SR tasks with asymmetric scale factors. It is observed that ArbSR includes the bilinear interpolation on the scale-specific paths which limits the capability of dynamic scale-aware filters. The multi-path learning strategy requires high computational and memory costs and bears the disadvantage of being complicated and limited when it comes to continuous scale factors. The scale factor is used for selecting the path and adjusting the weight according to the paths. The alternative is to treat the scale factor as the parameter of the function. LIIF [Chen *et al.*, 2021] first tries to represent an image as a continuous function that learns the continuous representation of images that follows the idea of implicit neural representations. However, LIIF does not make full use of the scale factor and the LR images, thus resulting in relatively low performance on $\times 2, \times 3, \times 4$ scale SR tasks.

It is a tipping point to combine the simplicity of the interpolation and the capability of neural networks. Concretely, the kernel weights are learned by a neural network which can be seen as a function. And the scale factor is treated as one of the inputs of the function. In this way, the kernel weights are varied with different situations of the scale factor without the heavy computation and memory cost required by multi-path learning strategies.

In this paper, we propose a learnable interpolation module that performs better than traditional methods while pre-

serving its simplicity. In the learnable interpolation module, a neural network is introduced to replace the typically fixed interpolation kernel function. To make full use of the scale factor, a scale-aware channel attention module is also proposed for more powerful feature extraction and feature correlation learning. It adjusts the weights of different feature channels with regard to the scale factor to extract informative features and boosts the performance of the model. Besides, the scale-aware channel module is carefully designed to integrate with the widely used residual module for wider application. These two modules are designed as plug-ins to be easily embedded into the mainstream SR models. Baseline networks equipped with our network achieve competitive performance compared with state-of-the-art SISR approaches on symmetric and asymmetric scale factors with almost negligible additional computation and memory costs.

Our contributions can be summarized as follows:

- We propose a learnable interpolation method that amplifies the image to any size in a unified network. It retains the simplicity of commonly-used interpolation methods and surpasses the performance of existing methods on scale-arbitrary SR tasks.
- We propose a scale-aware channel attention module. It dynamically adjusts the weights of the feature channels according to the input scale factor, contributing to better performance for scale-arbitrary SR tasks.
- We conduct extensive experiments on five benchmark datasets. The results show that the network equipped with our plug-in module improves the PSNR value by 0.13dB on average and up to 0.27dB over the existing models with only a slight increase in the number of parameters and computational cost.

2 Related Work

2.1 Learning-based Upsampling

To enhance the performance of traditional interpolation methods, several learning-based upsampling methods have been proposed. The standard learning-based upsampling methods are transposed convolution and sub-pixel convolution. The transposed convolution [Zeyde *et al.*, 2010] is essentially the opposite of a vanilla convolution. However, it tends to produce crosshatch artifacts due to zero padding. Also, the upsampled feature values are fixed and redundant. The sub-pixel convolution [Shi *et al.*, 2016] was proposed to circumvent this problem. It generates features with a number of extra channels by convolution and then reshapes the features to obtain the output image. In practice, it is difficult for the deeper networks to ignore the repeating artifacts produced by the sub-pixel convolution layer.

However, these methods can only handle fixed scale factor SR problems. To address this, some upsampling methods that can handle arbitrary scale factors have been proposed in recent years. [Hu *et al.*, 2019] first proposed a Meta-Upsample module to solve the scale-arbitrary SR problem based on meta-learning. [Wang *et al.*, 2021] proposed a scale-aware upsampling module, which is composed of bilinear sampling and conditional convolution and realizes image super-resolution of asymmetric scale factors. [Chen *et al.*, 2021] proposed the LIIF to learn a continuous image representation with a local implicit image function. LIIF achieves state-of-the-art performance on larger-scale tasks but has no superiority on smaller-scale tasks. In addition, several methods have been proposed for learning dynamic interpolation under certain circumstances. [Guo *et al.*, 2021] proposed dynamic interpolation that dynamically learns weights from input views. In contrast to these works, our learnable interpolation method replaces the fixed kernel function with a neural network to solve the scale-arbitrary SR problem.

2.2 Single Image Super-Resolution

Due to the rapid development of deep neural networks, CNN-based SISR approaches have a distinct advantage over traditional approaches. [Dong *et al.*, 2015] first proposed SRCNN for SISR, which uses three convolution layers for super-resolution. [Lim *et al.*, 2017] proposed a particularly deep network called EDSR, which removed the batch normalization layers and used residual scaling to stabilize the training process. [Zhang *et al.*, 2018b] proposed a residual dense network (RDN) that combined the advantages of residual blocks and dense connection blocks to improve SR performance. [Zhang *et al.*, 2018a] introduced the attention mechanism to the super-resolution task, and achieved significant improvement. Recently, [Liang *et al.*, 2021] proposed SwinIR, which used Swin Transformer [Liu *et al.*, 2021] and achieved state-of-the-art performance.

To deal with the problem of scale-arbitrary super-resolution, [Lim *et al.*, 2017] proposed a Multi-Scale Deep Super-Resolution (MDSR) to integrate multiple modules of different integer scale factors. However, MDSR cannot handle image super-resolution with non-integer scales. [Hu *et al.*, 2019] proposed Meta-SR, which utilizes the meta-upsample module to realize scale-arbitrary super-resolution. However, Meta-SR is also limited in that the horizontal and vertical scale factors must be consistent. Subsequently, [Wang *et al.*, 2021] proposed the ArbSR, which includes feature adaptation modules for scale perception and a scale-aware upsampling module. In ArbSR, the height and width are decoupled, and different features for SR are extracted from different scale factors in the feature extraction stage. In our work, we devise a scale-aware channel attention module that uses the mechanism of co-adaption [Kong *et al.*, 2021] and attention to extract suitable features with regard to different scale factors. The extracted features are fed into our learnable interpolation module, which computes the interpolation kernel and uses the interpolation kernel to interpolate the features to obtain the high-resolution counterparts.

3 Methods

3.1 Learnable Interpolation

Let f be the learnable interpolation function, and c denotes the number of feature channels, F_x and F_y denote the features before interpolation and after interpolation, of which sizes are $c \times h \times w$ and $c \times h' \times w'$, respectively. Remark that $h' > h$ and $w' > w$. The horizontal and vertical scale factors are $r_h = \frac{w'}{w}$

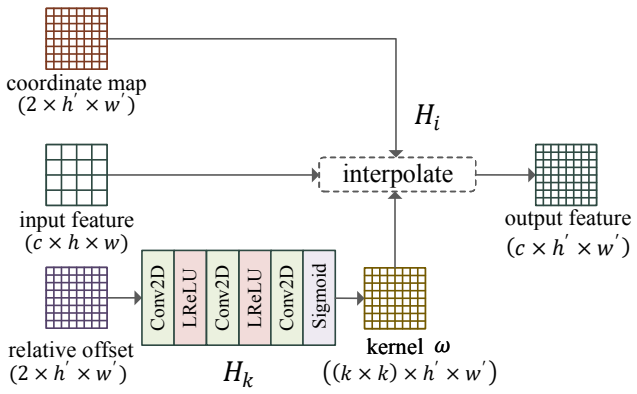

Figure 1: Overview of the learnable interpolation module. It adopts the neural network H_k to compute the interpolation kernel ω, and then upsamples the input feature F_x through H_i to produce the output F_y. See Equation (5), (6), (7), and (7) for the description of H_i.

and $r_v = \frac{h'}{h}$. Then the learnable interpolation problem can be formulated as:

$$F_y = f(F_x, r_h, r_v). \tag{1}$$

Similar to conventional interpolation methods, one may utilize the relative offset to compute the interpolation kernel. The relative offset tensor R with the size $2 \times h' \times w'$, can be seen as the mapping from high resolution (HR) space to low resolution (LR) space, and is defined as follows:

$$R(:, i, j) = \left(\left\{ \frac{i + 0.5}{r_h} \right\} - 0.5, \left\{ \frac{j + 0.5}{r_v} \right\} - 0.5 \right), \tag{2}$$

where $\{x\} = x - \lfloor x \rfloor$, and i, j are the pixel coordinates in HR space. To calculate the exact offset, a translation 0.5 in Equation (2) is used to represent the pixel center.

Rather than fixed kernel functions, in our learnable interpolation module, we utilize a neural network denoted by H_k, to compute the dynamic interpolation kernel,

$$\omega = H_k(R), \tag{3}$$

where ω is interpolation kernel of size $k \times k \times h' \times w'$ and k is the interpolation kernel size. Concretely, H_k consists of three convolution layers, two LeakyReLU activation layers [Maas *et al.*, 2013] and a sigmoid activation layer, whose network structure is depicted in Figure 1. $\omega(:, :, i', j')$ denotes that the (i', j') point in the HR space corresponding to the interpolation weight of the k nearest neighbors of the point (i, j) in the LR space.

To achieve better performance, we add a multi-head mechanism [Vaswani *et al.*, 2017] in the learnable interpolation module. Suppose m is the number of heads in the multi-head mechanism, and assume that m is a factor of c, *i.e.*, $m|c$. One can compute m groups of different interpolation kernels ω separately and get the global kernel $\hat{\omega}$ of size $m \times k \times k \times h' \times w'$ by concatenating kernels ω from m groups. $\hat{F}_x$ with the size $m \times \frac{c}{m} \times h \times w$ is obtained by partitioning the input feature F_x into m equal subsets.

Based on the above discussion, F_y in Equation (1) can be rewritten as:

$$F_y = H_i(\hat{F}_x, \hat{\omega}, r_h, r_v), \tag{4}$$

where H_i is the interpolation operation in our learnable interpolation module. Suppose (i', j') is the coordinate of a pixel in the HR space, let us explain H_i that computes the interpolated feature $F_y(i', j')$. Let $\bar{\omega}$ be the local kernel with the size $m \times k \times k$ corresponding to the pixel (i', j'). And it can be captured by indexing on the global kernel $\hat{\omega}$, *i.e.*,

$$\bar{\omega} = \hat{\omega}(:, :, :, i', j'). \tag{5}$$

For the given scale factor r_h, r_v, our learnable interpolation produces the corresponding pixel (i, j) in the LR space, *i.e.*,

$$i = \left\lfloor \frac{i' + 0.5}{r_h} \right\rfloor, \quad j = \left\lfloor \frac{j' + 0.5}{r_v} \right\rfloor. \tag{6}$$

To enrich the information for every pixel, we consider the k nearest neighbors of input feature $\hat{F}_x$. The concatenation of these features is defined as:

$$\bar{F}_x = \text{Concat}\left(\left\{ \hat{F}_x(:, :, i + p, j + q), -\left\lfloor \frac{k}{2} \right\rfloor \le p, q \le \left\lfloor \frac{k}{2} \right\rfloor \right\} \right),$$

where Concat refers to the concatenation of a set of vectors. By performing the above operations, the weighted sum of $\bar{F}_x$ over $\bar{\omega}$ is reformulated as the interpolated feature of size $m \times \frac{c}{m}$,

$$\bar{F}_y(p, q) = \sum_{k_1=0}^{k-1} \sum_{k_2=0}^{k-1} \bar{\omega}(p, k_1, k_2) \cdot \bar{F}_x(p, q, k_1, k_2), \tag{7}$$

where p, q are the traversal of the first two dimensions in $\bar{F}_x$, $p \in [0, m-1], q \in [0, \frac{c}{m} - 1]$. Notably, the final output F_y can be easily obtained by reshaping $\bar{F}_y$.

Intuitively, the goal of our learnable interpolation module is to support the interaction and the integration of aspects of CNN-based SISR methods and conventional interpolation methods, *i.e.*, the learnable interpolation module can perform as well as the CNN-based networks on the super-resolution task while preserving the flexibility of conventional interpolation methods. Our learnable interpolation module can provide suitable kernel functions for scale-arbitrary SR tasks by updating the parameters for the interpolation kernel functions through backpropagation. Benefiting from these appealing features, our proposed model can achieve considerable performance.

3.2 Scale-Aware Channel Attention

Since the degradation varies for different scale factors [Simonyan *et al.*, 2014; Kong *et al.*, 2021], it is necessary to take the scale factor into account for better performance. This finding inspires us to take advantage of feature recalibration in terms of different scale factors to improve the reconstruction performance. As demonstrated in Section 4.4, the features exhibit specificity for different scaling factors. To this end, we propose a scale-aware channel attention module by integrating the scale factor with channel attention.

In addition, it is essential to combine the scale-aware channel attention module with the baseline network. It can be noticed that a majority of feature extraction modules of super-resolution models contain residual structures for feature extraction. With the goal of wider applicability and better

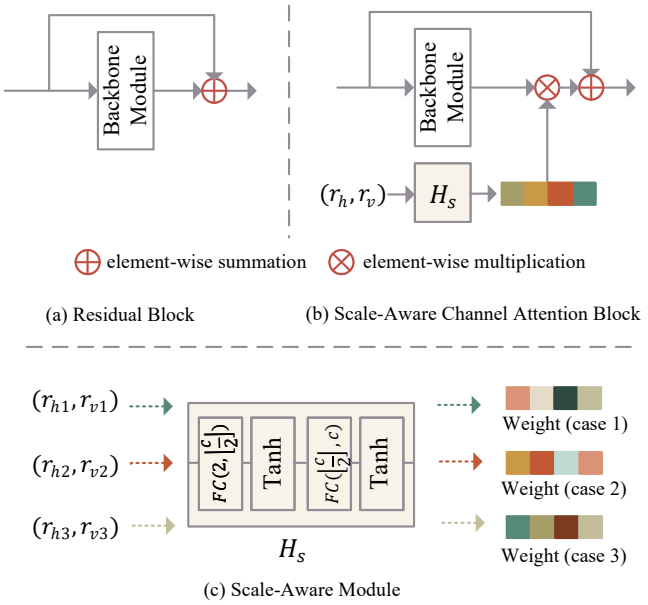

Figure 2: Overview of Scale-Aware Channel Attention block. (a) The original residual block. (b) Scale-aware channel attention block. (c) Scale-aware module. For a given scale factor, the scale-aware channel attention module generates adaptive channel weights to extract suitable features in the feature extraction stage.

performance, we combine the scale-aware channel attention module with residual structure.

We denote the input of the residual module as $F_i \in \mathbb{R}^{c\times h\times w}$, the output as $F_o \in \mathbb{R}^{c\times h\times w}$ and the residual backbone as H_b. The scale-aware channel attention module takes the scale factor into account and recalibrates features:

$$F_o = F_i + H_s(r_h, r_v) \cdot H_b(F_i), \tag{8}$$

where H_s is a neural network consisting of two fully connected layers with $\tanh$ activation function as shown in Figure 2. H_s can generate suitable channel weights in terms of different scaling factors. Thanks to its simple structure, the introduced parameters and additional computational cost are negligible.

3.3 Integration with SR architectures

The integration of the learnable interpolation module and the scale-aware channel attention into the existing networks is easy since the residual modules are widely used and the upsample module is indispensable for SR architectures. Designed as plug-ins, the learnable interpolation module replaces the original upsample module. Also, the scale-aware channel attention is integrated and jointly trained with the residual module.

As shown in Figure 3, the inference process is divided into two stages. In the feature extraction stage, the scale-aware channel attention module perceives the input scale factor to dynamically adjust the weights of different channels to extract features suitable for the super-resolution task of the current scale factor. In the upsampling stage, the learnable interpolation module uses the dynamically calculated kernel to

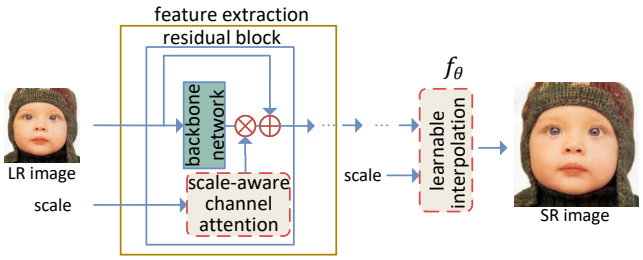

Figure 3: Integration with the existing network. For the given scale factor, the scale-aware channel attention module generates adaptive channel weights to extract suitable features in the feature extraction stage. Instead of the fixed kernel function, our proposed learnable interpolation module builds a DNN-type kernel function, which is able to yield the interpolation kernel from the relative offset.

interpolate the features extracted in the previous stage and obtain high-resolution images. In this way, we are capable of interpolating the features and obtaining the HR image output of the target size.

4 Experiments

4.1 Datasets and Metrics

Following Meta-SR [Hu *et al.*, 2019], we use the DIV2K [Timofte *et al.*, 2017] dataset as our training dataset. For testing, we evaluate our model on five standard benchmark datasets, *i.e.*, Set5 [Bevilacqua *et al.*, 2012], Set14 [Zeyde *et al.*, 2010], B100 [Martin *et al.*, 2001], Urban100 [Huang *et al.*, 2015] and Manga109 [Huang *et al.*, 2015]. The PSNR value [Wang *et al.*, 2004] on the Y channel of the transformed YCbCr color space is utilized to evaluate the performance of our method. Following the settings of previous works, we do the same border crop operation before calculating the evaluation metrics for a fair comparison.

4.2 Implementation Details

Our network uses the BasicSR [Wang *et al.*, 2018a] framework. In order to improve the efficiency of the learnable interpolation function, we use CUDA [NVIDIA *et al.*, 2020; Okuta *et al.*, 2017] Programming language to implement the above logic. The experiment is implemented on an NVIDIA RTX 3090 GPU with PyTorch [Paszke *et al.*, 2019].

We adopt two training strategies for the pretrain phase and the finetune phase [Park *et al.*, 2020]. During the pretrain phase, we specify several specific symmetric scale factor pairs (*i.e.*, $\times 2$, $\times 3$, $\times 4$) and generate LR training images to pretrain the model based on the selected scale factors. During the finetune phase, we select asymmetric scale factors uniformly and randomly in $[1.5, 4.5]$ to generate LR training images to finetune the model based on the pretrained model.

To stabilize the memory consumption during training, we fix the resolution of LR input patches since the extracted features are of the same size as LR input patches before upsampling. A pair of horizontal/vertical scale factors (r_h, r_v) is randomly selected for each batch during training. We randomly extract 16 ground-truth (GT) patches with the size of $48r_v \times 48r_h$ as a batch input. Then, the GT patches are downsampled into LR patches with the size of 48×48 by bicubic

	Params	FLOPs	Set5			Set14			B100			Urban100			Manga109		
	(M)	(G)	×2	×1.6	×1.55	×2	×1.5	×1.65	×2	×1.4	×1.85	×2	×1.9	×1.95	×2	×1.7	×1.95
Bicubic	-	-	33.66	36.10	36.24	30.24	32.87	31.83	29.56	32.95	30.11	26.88	27.25	27.05	30.80	32.91	31.12
EDSR-×2	40.7	166.8	38.19	40.39	40.71	33.95	37.10	35.95	32.36	36.79	33.02	32.95	33.06	32.69	39.18	40.88	39.13
ArbEDSR	-	-	38.19	40.64	40.94	34.05	37.51	36.22	32.37	36.92	33.23	33.02	33.61	33.30	39.22	41.20	39.24
LIEDSR	39.6	157.8	**38.26**	**40.74**	**41.03**	**34.09**	**37.52**	**36.29**	**32.40**	**36.93**	**33.27**	**33.11**	**33.75**	**33.43**	**39.27**	**41.32**	**39.56**
RDN-×2	22.1	90.6	38.24	40.51	40.53	34.01	37.24	36.10	32.34	36.83	33.15	32.89	33.05	32.79	39.18	41.06	39.31
Meta-RDN	22.4	97.2	38.23	40.66	40.94	34.03	37.52	36.24	32.35	36.93	33.21	33.03	33.60	33.26	39.31	41.33	39.60
ArbRDN	-	-	38.23	40.67	40.95	34.07	37.53	36.27	32.37	36.93	33.21	33.00	33.51	33.19	39.28	41.32	39.54
RDN-LIIF	21.3	112.6	38.16	40.61	41.00	34.06	37.54	36.35	32.26	36.86	33.07	32.90	33.53	33.22	39.19	41.29	39.49
LIRDN	22.1	90.3	**38.29**	**40.77**	**41.08**	**34.16**	**37.59**	**36.38**	**32.40**	**36.98**	**33.27**	**33.08**	**33.70**	**33.38**	**39.32**	**41.38**	**39.62**
RCAN-×2	15.4	62.8	38.27	40.53	40.77	34.12	37.23	36.08	32.40	36.86	33.16	33.18	33.17	32.84	39.42	41.15	39.39
Meta-RCAN	15.7	69.4	38.22	40.66	40.93	34.00	37.51	36.17	32.36	36.95	33.22	33.12	33.62	33.30	39.32	41.30	39.59
ArbRCAN	16.9	62.7	38.26	40.69	40.97	34.09	37.53	36.28	32.39	36.93	33.23	**33.14**	33.55	33.25	39.37	41.32	39.56
LIRCAN	15.8	62.4	**38.29**	**40.78**	**41.09**	**34.33**	**37.65**	**36.42**	**32.42**	**37.01**	**33.30**	33.13	**33.77**	**33.45**	**39.56**	**41.59**	**39.84**
			×3	×2.4	×2.75	×3	×2.8	×2.95	×3	×2.2	×2.15	×3	×2.3	×2.35	×3	×2.7	×2.55
Bicubic	-	-	30.39	32.41	31.06	27.55	27.84	27.46	27.21	28.88	29.12	24.46	25.91	25.72	26.95	27.77	28.27
EDSR-×3	43.7	179.1	34.68	36.45	35.35	30.53	30.90	30.49	29.27	31.38	31.78	28.82	31.13	30.91	34.19	35.18	35.75
ArbEDSR	-	-	34.73	**36.54**	35.34	30.61	31.04	30.56	29.30	31.46	31.70	28.90	31.36	31.11	34.28	35.40	36.06
LIEDSR	39.6	158.0	**34.76**	36.50	**35.38**	**30.68**	**31.15**	**30.67**	**29.34**	**31.51**	**31.75**	**29.03**	**31.55**	**31.29**	**34.37**	**35.53**	**36.18**
RDN-×3	22.3	91.4	34.71	36.46	35.27	30.57	30.88	30.53	29.26	31.30	31.65	28.80	31.25	31.07	34.13	35.41	36.00
Meta-RDN	22.4	106.4	34.73	**36.55**	35.33	30.58	30.97	30.57	29.30	31.41	31.69	28.93	31.33	31.13	34.40	35.58	36.21
ArbRDN	-	-	30.71	**36.55**	35.35	30.59	30.98	30.58	29.30	31.45	31.69	28.86	31.33	31.14	34.43	35.60	36.20
RDN-LIIF	21.3	141.0	34.70	36.48	35.40	30.57	31.10	30.70	29.21	31.41	31.61	28.81	31.32	31.09	34.15	35.45	36.14
LIRDN	22.1	90.5	**34.75**	36.50	**35.38**	**30.68**	**31.15**	**30.70**	**29.33**	**31.50**	**31.74**	**29.04**	**31.51**	**31.27**	**34.49**	**35.62**	**36.28**
RCAN-×3	15.6	63.5	34.76	36.51	35.31	30.62	30.90	30.53	29.31	31.31	31.68	29.01	31.34	31.15	34.42	35.50	36.06
Meta-RCAN	15.7	78.5	34.76	36.58	35.36	30.58	31.00	30.56	29.29	31.44	31.70	28.96	31.43	31.20	34.40	35.55	36.21
ArbRCAN	16.9	62.8	34.76	**36.59**	35.39	30.64	31.01	30.59	29.32	31.48	31.72	28.98	31.48	31.26	34.55	35.64	36.27
LIRCAN	15.8	62.6	**34.82**	36.57	**35.46**	**30.77**	**31.22**	**30.73**	**29.36**	**31.53**	**31.77**	**29.11**	**31.63**	**31.38**	**34.77**	**35.91**	**36.56**
			×4	×3.1	×3.25	×4	×3.2	×3.95	×4	×3.2	×3.55	×4	×3.7	×3.85	×4	×3.4	×3.65
Bicubic	-	-	28.42	29.89	29.21	26.00	26.98	25.68	25.96	26.91	26.32	23.14	23.38	23.14	24.89	25.97	25.41
EDSR-×4	43.1	205.8	32.47	34.25	33.35	28.81	29.95	28.63	27.73	28.84	28.25	26.65	27.06	26.69	31.04	32.51	31.79
ArbEDSR	-	-	32.51	34.48	33.92	28.83	30.07	28.72	27.74	28.91	28.30	26.62	27.12	26.73	31.26	32.90	32.14
LIEDSR	39.6	158.3	**32.59**	**34.52**	**34.00**	**28.91**	**30.18**	**28.81**	**27.79**	**28.95**	**28.35**	**26.81**	**27.34**	**27.02**	**31.38**	**33.00**	**32.24**
RDN-×4	22.3	93.1	32.47	34.36	33.91	28.81	30.01	28.69	27.72	28.85	28.25	26.61	27.17	26.83	31.00	32.70	31.99
Meta-RDN	22.4	119.1	32.49	34.42	33.93	28.86	30.06	28.75	27.75	28.90	28.31	26.70	27.24	26.91	31.34	33.02	32.24
ArbRDN	-	-	32.42	34.43	33.92	28.82	30.08	28.71	27.73	28.90	28.30	26.61	27.15	26.85	31.35	32.99	32.24
RDN-LIIF	21.3	180.6	32.54	34.49	**34.16**	28.85	30.12	28.93	27.70	28.85	28.29	26.67	27.19	26.92	31.15	32.84	32.09
LIRDN	22.1	90.8	**32.56**	**34.50**	33.97	**28.94**	**30.16**	**28.86**	**27.80**	**28.95**	**28.35**	**26.80**	**27.34**	**27.01**	**31.55**	**33.12**	**32.39**
RCAN-×4	15.6	65.3	32.63	34.37	33.92	28.85	30.00	28.72	27.75	28.86	28.27	26.75	27.20	26.89	31.20	32.76	32.04
Meta-RCAN	15.7	91.2	32.56	34.46	33.98	28.85	30.08	28.73	27.75	28.86	28.30	26.71	27.25	26.93	31.33	33.00	32.22
ArbRCAN	16.9	63.0	32.55	34.50	34.03	28.87	30.08	28.74	27.76	28.93	28.33	26.68	27.22	26.90	31.36	33.12	32.29
LIRCAN	15.8	62.9	**32.68**	**34.57**	**34.12**	**28.97**	**30.23**	**28.85**	**27.82**	**28.98**	**28.38**	**26.88**	**27.43**	**27.10**	**31.71**	**33.36**	**32.58**

Table 1: The PSNR (dB) results of our network with symmetric scale factors. **Bold** indicates the best result.

downsampling for training. In addition, random flipping is utilized for data augmentation.

We use EDSR [Lim *et al.*, 2017], RDN [Zhang *et al.*, 2018b] and RCAN [Zhang *et al.*, 2018a] as the baseline network and our plug-in module is embedded in the baseline networks to generate three scale-arbitrary networks, namely LIEDSR, LIRDN and LIRCAN. For each model, it is pretrained for 300K iterations and finetuned for 300K iterations. We set L_1 loss between SR results and HR images as the loss function. For optimization, we use Adam [Kingma and Ba, 2015] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. In order to stabilize the training process, we use the exponential moving average (EMA) strategy. The initial learning rate is set to 1×10^{-4} and halved at 200K iterations for both the pretrain and finetune phases.

4.3 Experimental Results

We compare our LISR with state-of-the-art methods including Meta-SR [Hu *et al.*, 2019], ArbSR [Wang *et al.*, 2021] and LIIF [Chen *et al.*, 2021]. We present the quantitative results including PSNR value, the total number of model parameters and FLOPs in Table 1 and 2, where FLOPs are measured with a 64×64 image as input, and the scale factor are consistent with ArbSR. Also, we present some representative qualitative comparison results in Figure 4 and 5. Since ArbSR does not provide the models and codes of ArbEDSR and ArbRDN, their parameters are not shown in the Tables.

Quantitative Results. It can be observed from Table 1 that our LIEDSR, LIRDN and LIRCAN perform significantly better than the baseline networks. For example, our LIEDSR has better results (40.74 vs 40.39 for $\times 1.6$ SR, 41.03 vs 40.71 for $\times 1.55$ SR) than EDSR on Set5.

Compared to Meta-SR, ArbSR and LIIF, our LISR network also generally achieves better PSNR performance. Among the 135 experiments with symmetric scale factors listed in Table 1, we get the highest PSNR metrics on 130 experiments. For example, our LIRCAN exceeds ArbRCAN by 0.35dB on PSNR on Manga109 with a scale factor of 4 and has 1.1M fewer parameters. Our LIRDN exceeds Meta-RDN by 0.35dB on PSNR on Manga109 with a scale factor of 4 and has 0.3M fewer parameters. Table 2 shows the experimental results of asymmetric scale factors. On the Urban100 dataset, the PSNR value of RCAN is 30.72 / 28.81 / 29.98 dB, the PSNR value of Meta-RCAN is 30.73 / 29.03 / 29.67 dB, and

	Params (M)	FLOPs (G)	Set5			Set14			B100			Urban100			Manga109		
			×1.5 / ×4	×1.5 / ×3.5	×1.6 / ×3.05	×4 / ×2	×3.5 / ×2	×3.5 / ×1.75	×4 / ×1.4	×1.5 / ×3	×3.5 / ×1.45	×1.6 / ×3	×1.6 / ×3.8	×3.55 / ×1.55	×2.5 / ×2	×2.8 / ×3.5	×3.35 / ×2.7
Bicubic	-	-	30.01	30.83	31.40	27.25	27.88	27.27	27.45	28.86	27.94	25.93	24.92	25.19	29.61	26.47	26.86
EDSR	43.1	183.9	33.95	34.89	35.59	30.29	30.91	31.36	29.33	31.24	29.96	30.61	28.77	29.23	37.08	32.99	33.46
ArbEDSR	-	-	34.32	35.33	36.02	30.51	31.15	31.46	29.52	31.38	30.20	31.06	29.32	29.98	37.70	33.54	34.16
LIEDSR	39.6	158.0	**34.48**	**35.36**	**36.12**	**30.62**	**31.27**	**31.58**	**29.63**	**31.42**	**30.27**	**31.25**	**29.54**	**30.17**	**37.81**	**33.67**	**34.30**
RDN	22.3	91.7	34.12	35.04	35.63	30.32	31.02	31.16	29.34	31.29	29.98	30.68	28.75	29.30	37.43	33.27	33.77
Meta-RDN	22.4	107.6	34.19	35.17	35.79	30.39	31.06	31.36	29.43	31.28	30.09	30.77	29.04	29.63	37.74	33.61	34.22
ArbRDN	-	-	34.31	35.26	35.98	30.47	31.12	31.42	29.52	31.36	30.19	31.02	29.23	29.91	37.88	33.74	34.36
RDN-LIIF	21.3	144.7	33.88	34.89	35.89	30.32	30.98	31.28	29.28	31.10	29.96	30.56	28.87	29.27	37.69	33.48	34.17
LIRDN	22.1	90.5	**34.45**	**35.38**	**36.08**	**30.66**	**31.30**	**31.56**	**29.63**	**31.42**	**30.26**	**31.21**	**29.49**	**30.16**	**37.92**	**33.78**	**34.40**
RCAN	15.6	63.9	34.14	35.05	35.67	30.35	31.02	31.21	29.35	31.30	29.98	30.72	28.81	29.34	37.48	33.31	33.82
Meta-RCAN	15.7	79.7	34.20	35.17	35.81	30.40	31.05	31.33	29.43	31.26	30.09	30.73	29.03	29.67	37.74	33.61	34.23
ArbRCAN	16.9	62.8	34.37	35.40	36.05	30.55	31.27	31.54	29.54	31.40	30.22	31.13	29.36	30.04	37.93	33.81	34.41
LIRCAN	15.8	62.3	**34.56**	**35.52**	**36.19**	**30.64**	**31.44**	**31.71**	**29.66**	**31.46**	**30.30**	**31.29**	**29.61**	**30.26**	**38.20**	**34.04**	**34.67**

Table 2: The PSNR (dB) results of our network with asymmetric scale factors. **Bold** indicates the best result.

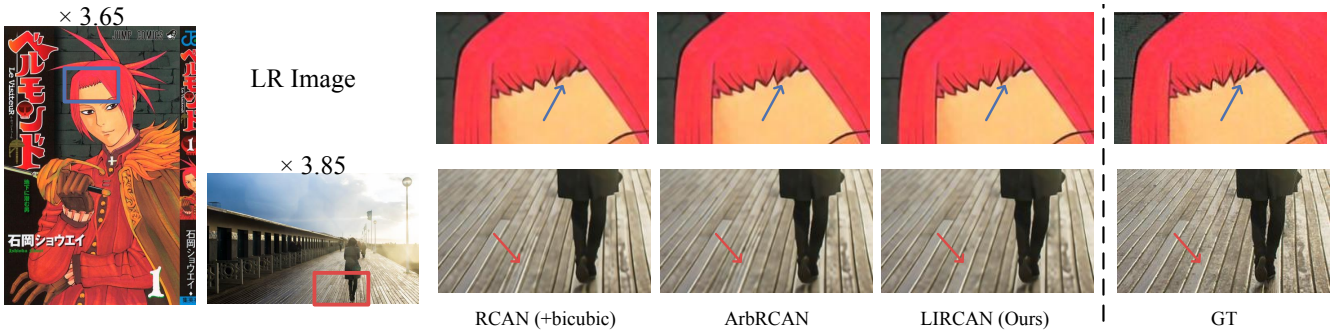

Figure 4: Visual comparison for SR of non-integer scale factors with RCAN as the baseline network. Zoom in for more details.

the PSNR value of ArbRCAN is 31.13 / 29.36 / 30.04 dB. And our LIRCAN model outperforms all their models with the PSNR value 31.29 / 29.61 / 30.26 dB.

Specifically, our LIRCAN network performs better on the Set14 dataset for $\times 2.8$ SR (31.20 dB), while the PSNR value of Meta-RCAN is 31.00 dB and that of ArbRCAN is 31.01 dB. Moreover, the number of model parameters added by our plug-in module is minimal among the three methods, and the number of model parameters of LIEDSR and LIRDN is even fewer than that of the baseline network.

Qualitative Results. We compare the visual results on image Belmondo from Manga109 dataset and img_032 from Urban100 dataset as shown in Figure 4. From the upper half of Figure 4, we can see that our network can handle complex textures better than RCAN and ArbRCAN. From the bottom part, it can be observed that RCAN and ArbRCAN models introduce unpleasant artifacts. Figure 5 shows the visual results on img_009 and img_037 from the Urban100 dataset. From the upper part of Figure 5, we can observe that our model produces fewer artifacts than other models. The bottom part of Figure 5 shows that the railway details generated by our model are more realistic and natural than that of RCAN and ArbRCAN. This demonstrates that our network achieves better visual quality with fewer artifacts and more realistic and natural textures than state-of-the-art approaches.

4.4 Ablation Studies

Ablation experiments are performed on Set5 and Manga109 to analyze the relative importance of each component in our method. We use EDSR as the baseline network and introduce two variants. In the case of variant 1, we replace the learnable interpolation module with bicubic interpolation and in the case of variant 2, we remove the scale-aware channel attention module. All variants are trained under the same settings.

The PSNR comparison results of LIEDSR and the two variants on Set5 are shown in Table 3. And the curves of PSNR on the Belmondo image from the Manga109 dataset for factors from 1.5 to 4.5 with a step of 0.01 are shown in Figure 6.

Ablation Study on Learnable Interpolation. From Table 3 and 6, we can conclude that in the case of variant 1, which replaces our learnable interpolation module with bicubic interpolation, is of relatively low PSNR value. When our learnable interpolation module is added, the improvement in PSNR results is significant (*e.g.*, 40.74 vs 37.37 for $\times 1.6$ SR, 34.48 vs 30.95 for $\frac{\times 1.5}{\times 4}$ SR). This shows that ordinary interpolation methods have poor performance due to their fixed and simple kernel functions. The learnable interpolation modifies the kernel function of ordinary interpolation, which increases the learnability and thus improves the performance.

Channel Importance Analysis. We first sequentially mask each channel of the feature, then compute their PSNR value. The experiment is conducted on the butterfly images in the Set5 dataset. It can be seen from Figure 7 that there are horizontal gradient stripes in the figure, which indicates that the feature importance value exhibits specificity with respect to

Model	learnable interpolation	scale-aware channel attention	$\times 1.6$	$\times 2$	$\times 2.75$	$\times 3$	$\times 3.25$	$\times 4$	$\frac{\times 1.5}{\times 4}$	$\frac{\times 1.5}{\times 3.5}$	$\frac{\times 1.6}{\times 3.05}$
EDSR	✗	✗	40.39	38.19	35.35	34.68	33.35	32.47	33.95	34.89	35.59
Variant 1	bicubic	✓	37.37	35.91	33.05	30.72	31.75	29.16	30.95	32.69	33.73
Variant 2	✓	✗	40.49	38.11	35.23	34.57	33.83	32.33	34.13	35.06	35.94
LIEDSR	✓	✓	**40.74**	**38.26**	**35.38**	**34.76**	**34.00**	**32.59**	**34.48**	**35.36**	**36.12**

Table 3: Quantitative ablation study on design choices of our network. **Bold** indicates the best result.

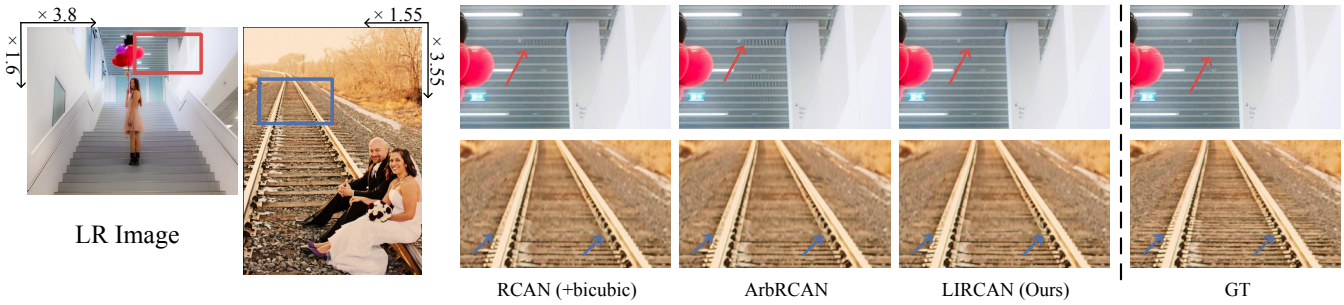

Figure 5: Visual comparison for SR of asymmetric scale factors with RCAN as the baseline network. Zoom in for more details.

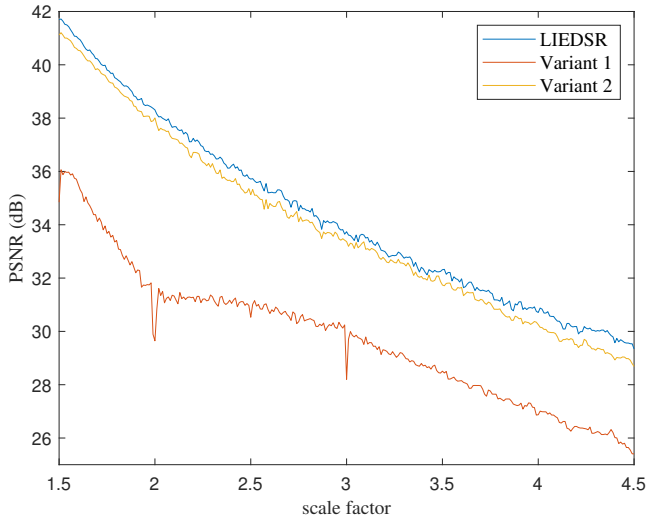

Figure 6: The horizontal axis represents the scale factor and the vertical axis represents the PSNR value of the model on the corresponding scale factor super-resolution task.

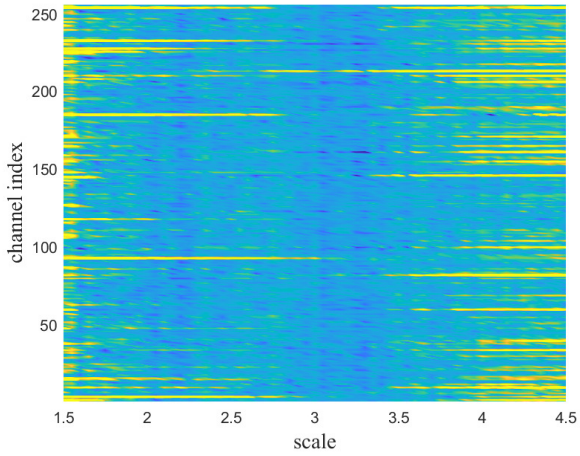

Figure 7: Feature channel importance heat map. The horizontal axis represents the scale factor, and the vertical axis represents the masked index of the channel. The feature channel is more influential on the PSNR results as the color becomes yellower and less important as it becomes bluer.

different scale factors. Therefore, for a specific scale factor, the scale-aware channel attention module can extract more suitable features.

Ablation Study on Scale-Aware Channel Attention. From Table 3 and Figure 6, we can see that our LIEDSR achieves higher PSNR values (*e.g.*, 35.38 vs 35.23 for $\times 2.75$ SR, 36.12 vs 35.94 for $\frac{\times 1.6}{\times 3.05}$ SR) compared with variant 2. As the only difference is that the scale-aware channel attention module is removed in the case of variant 2, we speculate that the scale-aware channel attention module can extract more suitable features with regard to the scale factor to improve the performance of the model.

5 Conclusion

This paper presents a novel plug-in module that includes a learnable interpolation module and scale-aware channel attention modules. Notably, the learnable interpolation module can amplify an image to any size in a unified network. Moreover, the scale-aware channel attention module can extract features suitable for any corresponding scale factor. Our plug-in module can be easily embedded in existing single-image super-resolution methods. Experiments demonstrate that baseline networks equipped with our plug-in module provide excellent quantitative and qualitative results for scale-arbitrary tasks with a relatively small increase in parameters and computational cost.

Acknowledgments

This work was supported in part by "Digital Silk Road" Shanghai International Joint Lab of Trustworthy Intelligent Software (Grant No. 22510750100), and Shanghai Trusted Industry Internet Software Collaborative Innovation Center.

References

[Aghighi, 2015] Hossein Aghighi. *Markov random field models for classification of remote sensing data and super-resolution mapping*. PhD thesis, University of New South Wales, Sydney, Australia, 2015.

[Ahn *et al.*, 2018] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part X*, volume 11214 of *Lecture Notes in Computer Science*, pages 256–272. Springer, 2018.

[Akhtar *et al.*, 2015] Naveed Akhtar, Faisal Shafait, and Ajmal S. Mian. Bayesian sparse representation for hyperspectral image super resolution. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3631–3640. IEEE Computer Society, 2015.

[Bevilacqua *et al.*, 2012] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie-Line Alberi-Morel. Low-complexity single-image super-resolution based on non-negative neighbor embedding. In Richard Bowden, John P. Collomosse, and Krystian Mikolajczyk, editors, *British Machine Vision Conference, BMVC 2012, Surrey, UK, September 3-7, 2012*, pages 1–10. BMVA Press, 2012.

[Chen *et al.*, 2021] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 8628–8638. Computer Vision Foundation / IEEE, 2021.

[De Boor, 1962] Carl De Boor. Bicubic spline interpolation. *Journal of mathematics and physics*, 41(1-4):212–218, 1962.

[Dong *et al.*, 2015] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *CoRR*, abs/1501.00092, 2015.

[Guo *et al.*, 2021] Mantang Guo, Jing Jin, Hui Liu, and Junhui Hou. Learning dynamic interpolation for extremely sparse light fields with wide baselines. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 2430–2439. IEEE, 2021.

[Hu *et al.*, 2019] Xuecai Hu, Haoyuan Mu, Xiangyu Zhang, Zilei Wang, Tieniu Tan, and Jian Sun. Meta-sr: A magnification-arbitrary network for super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 1575–1584. Computer Vision Foundation / IEEE, 2019.

[Huang *et al.*, 2015] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 5197–5206. IEEE Computer Society, 2015.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

[Kong *et al.*, 2021] Xiangtao Kong, Xina Liu, Jinjin Gu, Yu Qiao, and Chao Dong. Reflash dropout in image super-resolution. *CoRR*, abs/2112.12089, 2021.

[Liang *et al.*, 2021] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *IEEE/CVF International Conference on Computer Vision Workshops, ICCVW 2021, Montreal, BC, Canada, October 11-17, 2021*, pages 1833–1844. IEEE, 2021.

[Lim *et al.*, 2017] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPR Workshops*, pages 1132–1140. IEEE Computer Society, 2017.

[Lin *et al.*, 2008] Chung-Chi Lin, Ming-Hwa Sheu, Huann-Keng Chiang, Chishyan Liaw, and Zeng-Chuan Wu. The efficient VLSI design of BI-CUBIC convolution interpolation for digital image processing. In *International Symposium on Circuits and Systems (ISCAS 2008), 18-21 May 2008, Sheraton Seattle Hotel, Seattle, Washington, USA*, pages 480–483. IEEE, 2008.

[Liu *et al.*, 2021] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 9992–10002. IEEE, 2021.

[Maas *et al.*, 2013] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Citeseer, 2013.

[Martin *et al.*, 2001] David R. Martin, Charless C. Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, pages 416–425. IEEE Computer Society, 2001.

[NVIDIA *et al.*, 2020] NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. Cuda, release: 10.2.89, 2020.

[Okuta *et al.*, 2017] Ryosuke Okuta, Yuya Unno, Daisuke Nishino, Shohei Hido, and Crissman Loomis. Cupy: A numpy-compatible library for nvidia gpu calculations. In *Proceedings of Workshop on Machine Learning Systems*

(LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS), 2017.

[Park *et al.*, 2020] Seobin Park, Jinsu Yoo, Donghyeon Cho, Jiwon Kim, and Tae Hyun Kim. Fast adaptation to super-resolution networks via meta-learning. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXVII*, volume 12372 of *Lecture Notes in Computer Science*, pages 754–769. Springer, 2020.

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035, 2019.

[Shi *et al.*, 2016] Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, pages 1874–1883. IEEE Computer Society, 2016.

[Simonyan *et al.*, 2014] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*, 2014.

[Timofte *et al.*, 2017] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, Lei Zhang, Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee et al. NTIRE 2017 challenge on single image super-resolution: Methods and results. In *CVPR Workshops*, pages 1110–1121. IEEE Computer Society, 2017.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[Wang *et al.*, 2004] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, 2004.

[Wang *et al.*, 2018a] Xintao Wang, Ke Yu, Kelvin C.K. Chan, Chao Dong, and Chen Change Loy. BasicSR: Open source image and video restoration toolbox. https://github.com/xinntao/BasicSR, 2018.

[Wang *et al.*, 2018b] Yifan Wang, Federico Perazzi, Brian McWilliams, Alexander Sorkine-Hornung, Olga Sorkine-Hornung, and Christopher Schroers. A fully progressive approach to single-image super-resolution. In *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 864–873. Computer Vision Foundation / IEEE Computer Society, 2018.

[Wang *et al.*, 2021] Longguang Wang, Yingqian Wang, Zaiping Lin, Jungang Yang, Wei An, and Yulan Guo. Learning A single network for scale-arbitrary super-resolution. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 4781–4790. IEEE, 2021.

[Zeyde *et al.*, 2010] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces*, volume 6920 of *Lecture Notes in Computer Science*, pages 711–730. Springer, 2010.

[Zhang *et al.*, 2018a] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, volume 11211 of *Lecture Notes in Computer Science*, pages 294–310. Springer, 2018.

[Zhang *et al.*, 2018b] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2472–2481. Computer Vision Foundation / IEEE Computer Society, 2018.